Laborator 1

Acest laborator este destinat recapitularii conceptelor de baza din Tehnologia Web, mai jos aveti cateva link-uri utile peste care va invit sa va uitati:

Documentul laboratorului

Informatii de baza referitoare la protocolul HTTP

Istoria evolutiei protolului HTTP

Informatii utile despre protocolul SOAP

Exemplu de serviciu SOAP public

Comparatie intre servicii web si aplicatii web

Folosirea serviciilor web expuse printr-un API REST

Verificati daca aveti instalat utilitarul curl folosind comanda

```
curl --version
```

Instalare curl pe Windows

Instalare POSTMAN

Exemplu folosire curl folosind HTTP Verb GET

```
curl https://jsonplaceholder.typicode.com/todos/1
```

Exemple de API-uri publice pe care le folosim in cadrul laboratorului

Flickr

Virus total

Api Random

Api pentru prezenta Link pe discord

Cele mai populare API-uri 2022

Tema 1, cu predare in saptamana a doua - 02.03.2022

Create an application that provides results from at least three different web services. Use or implement a system that allows (at least): monitoring of running web services, concurrent

number of requests, logging requests/responses. Observation: The third web service must use the results from previous two.

Additional Information:

- the application should have both a client side and a server side, which means that a minimal web interface is required;
- the web interface will be provider by the server and it will be used to send the requests
- the communication logic between web services will be implemented in the web server component;
- logging requests/responses: each call will log at least the following information: request, response, latency (response time)
- monitoring of running web services: a /metrics API route should be provided by the web server, which aggregates data obtained through logging the api calls and the requests received in the web server;
- concurrent number of requests: create a script that sends a number of parallel requests in batches (e.g. 500 requests in batches of 50 parallel requests) and draws out some metrics about the behaviour of your API;

General observations:

- Any solution that matches the requirements is accepted (i.e. the logic or parts of it can be implemented on client side);
- In the evaluation process, the complexity of the used APIs will be taken into consideration;
- At least one of the used APIs should require an API key;
- Any form of secret (e.g. email password, secret key) should not be hardcoded in code. Try to at least store it in a configuration file;
- The homework can be implemented in any programming language.